COMPUTER SCIENCE A
SECTION I
Time—1 hour and 15 minutes
Number of questions—40
Percent of total score—50

Directions: Determine the answer to each of the following questions or incomplete statements, using the available space for any necessary scratch work. Then decide which is the best of the choices given and fill in the corresponding circle on the answer sheet. No credit will be given for anything written in the examination booklet. Do not spend too much time on any one problem.

Notes:
- Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
- Assume that declarations of variables and methods appear within the context of an enclosing class.
- Assume that method calls that are not prefixed with an object or class name and are not shown within a complete class definition appear within the context of an enclosing class.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.

1. Consider the following incomplete method, which is intended to return the number of integers that evenly divide the integer `inputVal`. Assume that `inputVal` is greater than 0.

```
public static int numDivisors(int inputVal)
{
  int count = 0;
  for (int k = 1; k <= inputVal; k++)
  {
    if ( /* condition */ )
    {
      count++;
    }
  }
  return count;
}
```

Which of the following can be used to replace `/* condition */` so that `numDivisors` will work as intended?

(A) `inputVal % k == 0`

(B) `k % inputVal == 0`

(C) `inputVal % k != 0`

(D) `inputVal / k == 0`

(E) `k / inputVal > 0`

2. Consider the following code segment.

```
for (int r = 3; r > 0; r--)
{
  int c;

  for (c = 1; c < r; c++)
  {
    System.out.print("-");
  }
  for (c = r ; c <= 3; c++)
  {
    System.out.print("*");
  }

  System.out.println();
}
```

What is printed as a result of executing the code segment?

(A)
```
--*
-**
***
```

(B)
```
*--
**-
***
```

(C)
```
***
-**
--*
```

(D)
```
***
**-
*--
```

(E)
```
--*
***
--*
```

3. Consider the following two classes.

```
public class A
{
  public void show()
  {
    System.out.print("A");
  }
}


public class B extends A
{
  public void show()
  {
    System.out.print("B");
  }
}
```

What is printed as a result of executing the following code segment?

```
A obj = new B();
obj.show();
```

(A) `A`

(B) `B`

(C) `AB`

(D) `BA`

(E) The code results in a runtime error.

4. Consider the following instance variable and method.

```
private int[] arr;

/** Precondition: arr.length > 0
 *  @return the largest value in array arr
 */
public int findMax()
{
  int maxVal = 0;

  for (int val : arr)
  {
    if (val > maxVal)
    {
      maxVal = val;
    }
  }

  return maxVal;
}
```

Method `findMax` is intended to return the largest value in the array `arr`. Which of the following best describes the conditions under which the method `findMax` will not work as intended?

(A) The largest value in `arr` occurs only once and is in `arr[0]`.

(B) The largest value in `arr` occurs only once and is in `arr[arr.length - 1]`.

(C) The largest value in `arr` is negative.

(D) The largest value in `arr` is zero.

(E) The largest value in `arr` occurs more than once.

5. Assume that `x` and `y` are `boolean` variables and have been properly initialized.

```
(x || y) && x
```

Which of the following always evaluates to the same value as the expression above?

(A) `x`

(B) `y`

(C) `x && y`

(D) `x || y`

(E) `x != y`

---

6. Consider the following method, which is intended to return `true` if at least one of the three strings `s1`, `s2`, or `s3` contains the substring "`art`". Otherwise, the method should return `false`.

```
public static boolean containsArt(String s1, String s2, String s3)
{

  String all = s1 + s2 + s3;

  return (all.indexOf("art") != -1);

}
```

Which of the following method calls demonstrates that the method does <u>not</u> work as intended?

(A) `containsArt("rattrap", "similar", "today")`

(B) `containsArt("start", "article", "Bart")`

(C) `containsArt("harm", "chortle", "crowbar")`

(D) `containsArt("matriculate", "carat", "arbitrary")`

(E) `containsArt("darkroom", "cartoon", "articulate")`

7. Consider the following code segment.

```
for (int outer = 1; outer <= 6; outer++)
{
  for (int inner = outer; inner <= 6; inner++)
  {
    if (inner % 2 == 0)
    {
      System.out.print(inner + "  ");
    }
  }
  System.out.println();
}
```

What will be printed as a result of executing the code segment?

(A)
```
2  4  6
4  6
6
```

(B)
```
2  4  6
2  4  6
2  4  6
```

(C)
```
2  4  6
2  4  6
4  6
4  6
6
6
```

(D)
```
2  4  6
2  4  6
2  4  6
2  4  6
2  4  6
2  4  6
```

(E)
```
2  4
2  4
4
4
```


GO ON TO THE NEXT PAGE.

8. Consider the following method.

```
public static int[] operation(int[][] matrix, int r, int c)
{
  int[] result = new int[matrix.length];

  for (int j = 0 ; j < matrix.length ; j++)
  {
    result[j] = matrix[r][j] * matrix[j][c];
  }
  return result;
}
```

The following code segment appears in another method in the same class.

```
int[][] mat = {{3, 2, 1, 4},
               {1, 2, 3, 4},
               {2, 2, 1, 2},
               {1, 1, 1, 1}};

int[] arr = operation(mat, 1, 2);
```

Which of the following represents the contents of `arr` as a result of executing the code segment?

(A) `{6, 4, 2, 4}`

(B) `{1, 6, 3, 4}`

(C) `{4, 3, 6, 1}`

(D) `{4, 4, 2, 2}`

(E) `{2, 2, 4, 4}`

9. A pair of number cubes is used in a game of chance. Each number cube has six sides, numbered from 1 to 6, inclusive, and there is an equal probability for each of the numbers to appear on the top side (indicating the cube's value) when the number cube is rolled. The following incomplete statement appears in a program that computes the sum of the values produced by rolling two number cubes.

```
int sum = /* missing code */ ;
```

Which of the following replacements for `/* missing code */` would best simulate the value produced as a result of rolling two number cubes?

(A) `2 * (int) (Math.random() * 6)`

(B) `2 * (int) (Math.random() * 7)`

(C) `(int) (Math.random() * 6) + (int) (Math.random() * 6)`

(D) `(int) (Math.random() * 13)`

(E) `2 + (int) (Math.random() * 6) + (int) (Math.random() * 6)`

10. Consider the following interface and class declarations.

```
public interface Student
{  /* implementation not shown */  }

public class Athlete
{  /* implementation not shown */  }

public class TennisPlayer extends Athlete implements Student
{  /* implementation not shown */  }
```

Assume that each class has a zero-parameter constructor. Which of the following is NOT a valid declaration?

(A) `Student a = new TennisPlayer();`

(B) `TennisPlayer b = new TennisPlayer();`

(C) `Athlete c = new TennisPlayer();`

(D) `Student d = new Athlete();`

(E) `Athlete e = new Athlete();`

11. Consider the following method.

```
public static boolean mystery(String str)
{
  String temp = "";

  for (int k = str.length(); k > 0; k--)
  {
    temp = temp + str.substring(k - 1, k);
  }

  return temp.equals(str);
}
```

Which of the following calls to `mystery` will return `true`?

(A) `mystery("no")`

(B) `mystery("on")`

(C) `mystery("nnoo")`

(D) `mystery("nono")`

(E) `mystery("noon")`

12. Assume that `x` and `y` are `boolean` variables and have been properly initialized.

```
(x && y) && !(x || y)
```

Which of the following best describes the result of evaluating the expression above?

(A) `true` always

(B) `false` always

(C) `true` only when `x` is `true` and `y` is `true`

(D) `true` only when `x` and `y` have the same value

(E) `true` only when `x` and `y` have different values

13. Consider the following instance variable and method.

```
private int[] numbers;

public void mystery(int x)
{
  for (int k = 1; k < numbers.length; k = k + x)
  {
    numbers[k] = numbers[k - 1] + x;
  }
}
```

Assume that `numbers` has been initialized with the following values.

```
{17, 34, 21, 42, 15, 69, 48, 25, 39}
```

Which of the following represents the order of the values in `numbers` as a result of the call `mystery(3)` ?

(A) `{17, 20, 21, 42, 45, 69, 48, 51, 39}`

(B) `{17, 20, 23, 26, 29, 32, 35, 38, 41}`

(C) `{17, 37, 21, 42, 18, 69, 48, 28, 39}`

(D) `{20, 23, 21, 42, 45, 69, 51, 54, 39}`

(E) `{20, 34, 21, 45, 15, 69, 51, 25, 39}`

14. Consider the following method, `biggest`, which is intended to return the greatest of three integers. It does not always work as intended.

```
public static int biggest(int a, int b, int c)
{
  if ((a > b) && (a > c))
  {
    return a;
  }
  else if ((b > a) && (b > c))
  {
    return b;
  }
  else
  {
    return c;
  }
}
```

Which of the following best describes the error in the method?

(A) `biggest` always returns the value of `a`.

(B) `biggest` may not work correctly when `c` has the greatest value.

(C) `biggest` may not work correctly when `a` and `b` have equal values.

(D) `biggest` may not work correctly when `a` and `c` have equal values.

(E) `biggest` may not work correctly when `b` and `c` have equal values.

15. Consider the following method.

```
public static void showMe(int arg)
{
  if (arg < 10)
  {
    showMe(arg + 1);
  }
  else
  {
    System.out.print(arg + " ");
  }
}
```

What will be printed as a result of the call `showMe(0)` ?

(A) `10`

(B) `11`

(C) `0 1 2 3 4 5 6 7 8 9`

(D) `9 8 7 6 5 4 3 2 1 0`

(E) `0 1 2 3 4 5 6 7 8 9 10`

16. Consider the following method.

```
/** Precondition: values has at least one row */
public static int calculate(int[][] values)
{
  int found = values[0][0];
  int result = 0;
  for (int[] row : values)
  {
    for (int y = 0; y < row.length; y++)
    {
      if (row[y] > found)
      {
        found = row[y];
        result = y;
      }
    }
  }
  return result;
}
```

Which of the following best describes what is returned by the `calculate` method?

(A) The largest value in the two-dimensional array

(B) The smallest value in the two-dimensional array

(C) The row index of an element with the largest value in the two-dimensional array

(D) The row index of an element with the smallest value in the two-dimensional array

(E) The column index of an element with the largest value in the two-dimensional array

17. Consider the following method.

```
//* Precondition: num > 0 */
public static int doWhat(int num)
{
  int var = 0;

  for (int loop = 1; loop <= num; loop = loop + 2)
  {
    var += loop;
  }

  return var;
}
```

Which of the following best describes the value returned from a call to `doWhat` ?

(A) `num`

(B) The sum of all integers between `1` and `num`, inclusive

(C) The sum of all even integers between `1` and `num`, inclusive

(D) The sum of all odd integers between `1` and `num`, inclusive

(E) No value is returned because of an infinite loop.

---

18. What is printed as a result of executing the following statement?

```
System.out.println(404 / 10 * 10 + 1);
```

(A) `4`

(B) 5

(C) `41`

(D) `401`

(E) `405`

19. Consider the following code segment.

```
int x = 1;
while ( /* condition */ )
{
  if (x % 2 == 0)
  {
    System.out.print(x + " ");
  }
  x = x + 2;
}
```

The following conditions have been proposed to replace `/* condition */` in the code segment.

I. `x < 0`

II. `x <= 1`

III. `x < 10`

For which of the conditions will nothing be printed?

(A) I only

(B) II only

(C) I and II only

(D) I and III only

(E) I, II, and III

20. Consider the following method.

```
/** Precondition: arr.length > 0 */
public static int mystery(int[] arr)
{
  int index = 0;
  int count = 0;
  int m = -1;

  for (int outer = 0; outer < arr.length; outer++)
  {
    count = 0;
    for (int inner = outer + 1; inner < arr.length; inner++)
    {
      if (arr[outer] == arr[inner])
      {
        count++;
      }
    }

    if (count > m)
    {
      index = outer;
      m = count;
    }
  }

  return index;
}
```

Assume that `nums` has been declared and initialized as an array of integer values. Which of the following best describes the value returned by the call `mystery(nums)` ?

(A) The maximum value that occurs in `nums`

(B) An index of the maximum value that occurs in `nums`

(C) The number of times that the maximum value occurs in `nums`

(D) A value that occurs most often in `nums`

(E) An index of a value that occurs most often in `nums`

GO ON TO THE NEXT PAGE.

21. Consider the following recursive method.

```
public static void whatsItDo(String str)
{
  int len = str.length();
  if (len > 1)
  {
    String temp = str.substring(0, len - 1);
    System.out.println(temp);
    whatsItDo(temp);
  }
}
```

What is printed as a result of the call `whatsItDo("WATCH")` ?

(A)
```
H
```

(B)
```
WATC
```

(C)
```
ATCH
ATC
AT
A
```

(D)
```
WATC
WAT
WA
W
```

(E)
```
WATCH
WATC
WAT
WA
```

22. Consider the following definition.

```
int[][] numbers = {{1, 2, 3},
                   {4, 5, 6}};
```

Which of the following code segments produces the output `123456` ?

(A)
```
for (int[] row : numbers)
{
  for (int n : row)
  {
    System.out.print(n);
  }
}
```

(B)
```
for (int[] row : numbers)
{
  for (int n : row)
  {
    System.out.print(row[n]);
  }
}
```

(C)
```
for (int rc = 0; rc < numbers.length; rc++)
{
  System.out.print(numbers[rc]);
}
```

(D)
```
for (int r = 0; r < numbers[0].length; r++)
{
  for (int c = 0; c < numbers.length; c++)
  {
    System.out.print(numbers[r][c]);
  }
}
```

(E)
```
for (int c = 0; c < numbers[0].length; c++)
{
  for (int r = 0; r < numbers.length; r++)
  {
    System.out.print(numbers[r][c]);
  }
}
```

23. Consider the following code segment from an insertion sort program.

```
for (int j = 1; j < arr.length; j++)
{
  int insertItem = arr[j];
  int k = j - 1;

  while (k >= 0 && insertItem < arr[k])
  {
    arr[k + 1] = arr[k];
    k--;
  }

  arr[k + 1] = insertItem;

  /* end of for loop */
}
```

Assume that array `arr` has been defined and initialized with the values `{5, 4, 3, 2, 1}`. What are the values in array `arr` after two passes of the `for` loop (i.e., when `j = 2` at the point indicated by `/* end of for loop */`) ?

(A) `{2, 3, 4, 5, 1}`

(B) `{3, 2, 1, 4, 5}`

(C) `{3, 4, 5, 2, 1}`

(D) `{3, 5, 2, 3, 1}`

(E) `{5, 3, 4, 2, 1}`

24. Consider the following class.

```
public class SomeMethods
{
  public void one(int first)
  {  /* implementation not shown */  }

  public void one(int first, int second)
  {  /* implementation not shown */  }

  public void one(int first, String second)
  {  /* implementation not shown */  }
}
```

Which of the following methods can be added to the `SomeMethods` class without causing a compile-time error?

I.
```
public void one(int value)
{  /* implementation not shown */  }
```

II.
```
public void one(String first, int second)
{  /* implementation not shown */  }
```

III.
```
public void one(int first, int second, int third)
{  /* implementation not shown */  }
```

(A) I only

(B) I and II only

(C) I and III only

(D) II and III only

(E) I, II, and III

25. Consider the following code segment.

```
int count = 0;

for (int x = 0; x < 4; x++)
{
  for (int y = x; y < 4; y++)
  {
    count++;
  }
}
System.out.println(count);
```

What is printed as a result of executing the code segment?

(A) 4

(B) 8

(C) 10

(D) 16

(E) 20

GO ON TO THE NEXT PAGE.

26. Consider the following two methods, which appear within a single class.

```
public static void changeIt(int[] arr, int val, String word)
{
  arr = new int[5];
  val = 0;
  word = word.substring(0, 5);

  for (int k = 0; k < arr.length; k++)
  {
    arr[k] = 0;
  }
}


public static void start()
{
  int[] nums = {1, 2, 3, 4, 5};
  int value = 6;
  String name = "blackboard";

  changeIt(nums, value, name);

  for (int k = 0; k < nums.length; k++)
  {
    System.out.print(nums[k] + " ");
  }

  System.out.print(value + " ");
  System.out.print(name);
}
```

What is printed as a result of the call `start()` ?

(A) `0 0 0 0 0 0 black`

(B) `0 0 0 0 0 6 blackboard`

(C) `1 2 3 4 5 6 black`

(D) `1 2 3 4 5 0 black`

(E) `1 2 3 4 5 6 blackboard`

Questions 27-28 refer to the following information.

Consider the following `sort` method. This method correctly sorts the elements of array `data` into increasing order.

```
public static void sort(int[] data)
{
  for (int j = 0; j < data.length - 1; j++)
  {
    int m = j;
    for (int k = j + 1; k < data.length; k++)
    {
      if (data[k] < data[m])     /* Compare values */
      {
        m = k;
      }
    }
    int temp = data[m];          /* Assign to temp */
    data[m] = data[j];
    data[j] = temp;

    /* End of outer loop */
  }
}
```

27. Assume that `sort` is called with the array `{6, 3, 2, 5, 4, 1}`. What will the value of `data` be after three passes of the outer loop (i.e., when `j = 2` at the point indicated by `/* End of outer loop */`) ?

(A) `{1, 2, 3, 4, 5, 6}`

(B) `{1, 2, 3, 5, 4, 6}`

(C) `{1, 2, 3, 6, 5, 4}`

(D) `{1, 3, 2, 4, 5, 6}`

(E) `{1, 3, 2, 5, 4, 6}`

---

28. Assume that `sort` is called with the array `{1, 2, 3, 4, 5, 6}`. How many times will the expression indicated by `/* Compare values */` and the statement indicated by `/* Assign to temp */` execute?

	Compare values	*Assign to temp*
(A)	15	0
(B)	15	5
(C)	15	6
(D)	21	5
(E)	21	6

29. Consider the following recursive method.

```
/** Precondition: num ≥ 0 */
public static int what(int num)
{
  if (num < 10)
  {
    return 1;
  }
  else
  {
    return 1 + what(num / 10);
  }
}
```

Assume that `int val` has been declared and initialized with a value that satisfies the precondition of the method. Which of the following best describes the value returned by the call `what(val)` ?

(A) The number of digits in the decimal representation of `val` is returned.

(B) The sum of the digits in the decimal representation of `val` is returned.

(C) Nothing is returned. A run-time error occurs because of infinite recursion.

(D) The value 1 is returned.

(E) The value `val/10` is returned.

30. The price per box of ink pens advertised in an office supply catalog is based on the number of boxes ordered. The following table shows the pricing.

Number of Boxes	Price per Box
1 up to but not including 5	$5.00
5 up to but not including 10	$3.00
10 or more	$1.50

The following incomplete method is intended to return the total cost of an order based on the value of the parameter `numBoxes`.

```
/** Precondition: numBoxes > 0 */
public static double getCost(int numBoxes)
{
  double totalCost = 0.0;

  /* missing code */

  return totalCost;
}
```

Which of the following code segments can be used to replace `/* missing code */` so that method `getCost` will work as intended?

I.
```
if (numBoxes >= 10)
{
  totalCost = numBoxes * 1.50;
}
if (numBoxes >= 5)
{
  totalCost = numBoxes * 3.00;
}
if (numBoxes > 0)
{
  totalCost = numBoxes * 5.00;
}
```

II.
```
if (numBoxes >= 10)
{
  totalCost = numBoxes * 1.50;
}
else if (numBoxes >= 5)
{
  totalCost = numBoxes * 3.00;
}
else
{
  totalCost = numBoxes * 5.00;
}
```

III.
```
if (numBoxes > 0)
{
  totalCost = numBoxes * 5.00;
}
else if (numBoxes >= 5)
{
  totalCost = numBoxes * 3.00;
}
else if (numBoxes >= 10)
{
  totalCost = numBoxes * 1.50;
}
```

(A) I only

(B) II only

(C) III only

(D) I and II

(E) II and III

31. Consider the following code segment.

```
String[][] board = new String[5][5];

for (int row = 0; row < 5; row++)
{
  for (int col = 0; col < 5; col++)
  {
    board[row][col] = "O";
  }
}

for (int val = 0; val < 5; val++)
{
  if (val % 2 == 1)
  {
    int row = val;
    int col = 0;
    while (col < 5 && row >= 0)
    {
      board[row][col] = "X";
      col++;
      row--;
    }
  }
}
```

Which of the following represents `board` after this code segment is executed?

(A)

	0	1	2	3	4
0	X	O	X	O	X
1	O	X	O	X	O
2	X	O	X	O	X
3	O	X	O	X	O
4	X	O	X	O	X

(B)

	0	1	2	3	4
0	O	X	O	X	O
1	X	O	X	O	X
2	O	X	O	X	O
3	X	O	X	O	X
4	O	X	O	X	O

(C)

	0	1	2	3	4
0	X	O	O	O	X
1	O	X	O	X	O
2	O	O	X	O	O
3	O	X	O	X	O
4	X	O	O	O	X

(D)

	0	1	2	3	4
0	O	X	O	O	O
1	O	O	X	O	O
2	X	O	O	X	O
3	O	X	O	O	X
4	O	O	X	O	O

(E)

	0	1	2	3	4
0	O	X	O	X	O
1	X	O	X	O	O
2	O	X	O	O	O
3	X	O	O	O	O
4	O	O	O	O	O

32. Consider the following class declaration.

```
public class StudentInfo
{
  private String major;
  private int age;

  public String getMajor()
  {  return major;  }

  public int getAge()
  {  return age;  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

The following instance variable and method appear in another class.

```
private List<StudentInfo> students;

/** @return the average age of students with the given major;
 *          -1.0 if no such students exist
 */
public double averageAgeInMajor(String theMajor)
{
  double sum = 0.0;
  int count = 0;
  for (StudentInfo k : students)
  {
    /* missing code */
  }

  if (count > 0)
  {
    return sum / count;
  }
  else
  {
    return -1.0;
  }
}
```

Which of the following could be used to replace `/* missing code */` so that `averageAgeInMajor` will compile without error?

(A)
```
if (theMajor.equals(k.major))
{
  sum += k.age;
  count++;
}
```

(B)
```
if (theMajor.equals(k.getMajor()))
{
  sum += k.getAge();
  count++;
}
```

(C)
```
if (theMajor.equals(k.major))
{
  sum += k.getAge();
  count++;
}
```

(D)
```
if (theMajor.equals(students[k].getMajor()))
{
  sum += students[k].getAge();
  count++;
}
```

(E)
```
if (theMajor.equals(getMajor(k)))
{
  sum += getAge(k);
  count++;
}
```

---

33. Which of the following statements regarding interfaces is FALSE?

(A) All methods in an interface are public.

(B) An interface cannot be instantiated.

(C) An interface can declare an instance variable.

(D) A non-abstract class can implement an interface.

(E) An abstract class can implement an interface.

34. Consider the problem of finding the maximum value in an array of integers. The following code segments are proposed solutions to the problem. Assume that the variable `arr` has been defined as an array of `int` values and has been initialized with one or more values.

I.
```
int max = Integer.MIN_VALUE;
for (int value : arr)
{
  if (max < value)
  {
    max = value;
  }
}
```

II.
```
int max = 0;
boolean first = true;
for (int value : arr)
{
  if (first)
  {
    max = value;
    first = false;
  }
  else if (max < value)
  {
    max = value;
  }
}
```

III.
```
int max = arr[0];
for (int k = 1; k < arr.length; k++)
{
  if (max < arr[k])
  {
    max = arr[k];
  }
}
```

Which of the code segments will always correctly assign the maximum element of the array to the variable `max` ?

(A) I only

(B) II only

(C) III only

(D) II and III only

(E) I, II, and III

35. Consider the following instance variable and method. Method `wordsWithCommas` is intended to return a string containing all the words in `listOfWords` separated by commas and enclosed in braces.
For example, if `listOfWords` contains `["one", "two", "three"]`, the string returned by the call `wordsWithCommas()` should be `"{one, two, three}"`.

```
private List<String> listOfWords;

public String wordsWithCommas()
{
  String result = "{";

  int sizeOfList = /* expression */ ;

  for (int k = 0; k < sizeOfList; k++)
  {
    result = result + listOfWords.get(k);

    if ( /* condition */ )
    {
      result = result + ", ";
    }
  }

  result = result + "}";
  return result;
}
```

Which of the following can be used to replace `/* expression */` and `/* condition */` so that `wordsWithCommas` will work as intended?

	`/* expression */`	`/* condition */`
(A)	`listOfWords.size() - 1`	`k != 0`
(B)	`listOfWords.size()`	`k != 0`
(C)	`listOfWords.size() - 1`	`k != sizeOfList - 1`
(D)	`listOfWords.size()`	`k != sizeOfList - 1`
(E)	`result.length()`	`k != 0`

Questions 36-37 refer to the following information.

Consider the following `binarySearch` method. The method correctly performs a binary search.

```
/** Precondition: data is sorted in increasing order. */
public static int binarySearch(int[] data, int target)
{
  int start = 0;
  int end = data.length - 1;
  while (start <= end)
  {
    int mid = (start + end) / 2;      /* Calculate midpoint */
    if (target < data[mid])
    {
      end = mid - 1;
    }
    else if (target > data[mid])
    {
      start = mid + 1;
    }
    else
    {
      return mid;
    }
  }
  return -1;
}
```

36. Consider the following code segment.

```
int[] values = {1, 2, 3, 4, 5, 8, 8, 8};
int target = 8;
```

What value is returned by the call `binarySearch(values, target)` ?

(A) -1

(B) 3

(C) 5

(D) 6

(E) 8

---

37. Suppose the `binarySearch` method is called with an array containing 2,000 elements sorted in increasing order. What is the maximum number of times that the statement indicated by `/* Calculate midpoint */` could execute?

(A) 2,000

(B) 1,000

(C) 20

(D) 11

(E) 1

38. Consider the following incomplete method that is intended to return a string formed by concatenating elements from the parameter `words`. The elements to be concatenated start with `startIndex` and continue through the last element of `words` and should appear in reverse order in the resulting string.

```
/** Precondition: words.length > 0;
 *                startIndex >= 0
 */
public static String concatWords(String[] words, int startIndex)
{
  String result = "";

  /* missing code */

  return result;
}
```

For example, the following code segment uses a call to the `concatWords` method.

```
String[] things = {"Bear", "Apple", "Gorilla", "House", "Car"};
System.out.println(concatWords(things, 2));
```

When the code segment is executed, the string `"CarHouseGorilla"` is printed.

The following three code segments have been proposed as replacements for `/* missing code */`.

I.
```
for (int k = startIndex; k < words.length; k++)
{
  result += words[k] + words[words.length - k - 1];
}
```

II.
```
int k = words.length - 1;
while (k >= startIndex)
{
  result += words[k];
  k--;
}
```

III.
```
String[] temp = new String[words.length];
for (int k = 0; k <= words.length / 2; k++)
{
  temp[k] = words[words.length - k - 1];
  temp[words.length - k - 1] = words[k];
}

for (int k = 0; k < temp.length - startIndex; k++)
{
  result += temp[k];
}
```

Which of these code segments can be used to replace `/* missing code */` so that `concatWords` will work as intended?

(A) I only

(B) II only

(C) III only

(D) I and II

(E) II and III

39. Consider the following method.

```
/** Precondition: 0 < numVals <= nums.length */
public static int mystery(int[] nums, int v, int numVals)
{
  int k = 0;

  if (v == nums[numVals - 1])
  {
    k = 1;
  }

  if (numVals == 1)
  {
    return k;
  }
  else
  {
    return k + mystery(nums, v, numVals - 1);
  }
}
```

Which of the following best describes what the call `mystery(numbers, val, numbers.length)` does? You may assume that variables `numbers` and `val` have been declared and initialized.

(A) Returns 1 if the last element in `numbers` is equal to `val`; otherwise, returns 0

(B) Returns the index of the last element in `numbers` that is equal to `val`

(C) Returns the number of elements in `numbers` that are equal to `val`

(D) Returns the number of elements in `numbers` that are not equal to `val`

(E) Returns the maximum number of adjacent elements that are not equal to `val`

40. Consider the following code segment.

```
List<String> students = new ArrayList<String>();

students.add("Alex");
students.add("Bob");
students.add("Carl");

for (int k = 0; k < students.size(); k++)
{
  System.out.print(students.set(k, "Alex") + "   ");
}

System.out.println();

for (String str : students)
{
  System.out.print(str + "   ");
}
```

What is printed as a result of executing the code segment?

(A)
```
Alex  Alex  Alex
Alex  Alex  Alex
```

(B)
```
Alex  Alex  Alex
Alex  Bob  Carl
```

(C)
```
Alex  Bob  Carl
Alex  Alex  Alex
```

(D)
```
Alex  Bob  Carl
Alex  Bob  Carl
```

(E) Nothing is printed because the first print statement will cause a runtime exception to be thrown.